Breast Cancer Classification

Breast cancer (BC) is one of the most common cancers among women worldwide, representing most new cancer cases and cancer-related deaths according to global statistics, making it a significant public health problem in today's society.

Objective

The early diagnosis of BC can improve the prognosis and chance of survival significantly, as it can promote timely clinical treatment to patients. Further accurate classification of benign tumors can prevent patients undergoing unnecessary treatments. Thus, the correct diagnosis of BC and classification of patients into malignant or benign groups is the subject of much research. Because of its unique advantages in critical features detection from complex BC datasets, machine learning (ML) is widely recognized as the methodology of choice in BC pattern classification and forecast modelling.

The Data : UCI Machine Learning Repository for BC dataset

The dataset used in this story is publicly available and was created by Dr. William H. Wolberg, physician at the University of Wisconsin Hospital at Madison, Wisconsin, USA. To create the dataset Dr. Wolberg used fluid samples, taken from patients with solid breast masses and an easy-to-use graphical computer program called Xcyt, which is capable of perform the analysis of cytological features based on a digital scan. The program uses a curve-fitting algorithm, to compute ten features from each one of the cells in the sample, then it calculates the mean value, extreme value and standard error of each feature for the image, returning a 30 real-valuated vector

Data Wrangling

We conduct the following steps to clean up data and pick useful features.

1. Search for missing values:
   - **Unnamed: 32** feature includes NaN, so we do not need it.

2. Drop duplicates:
   - Didn't find any duplicates.

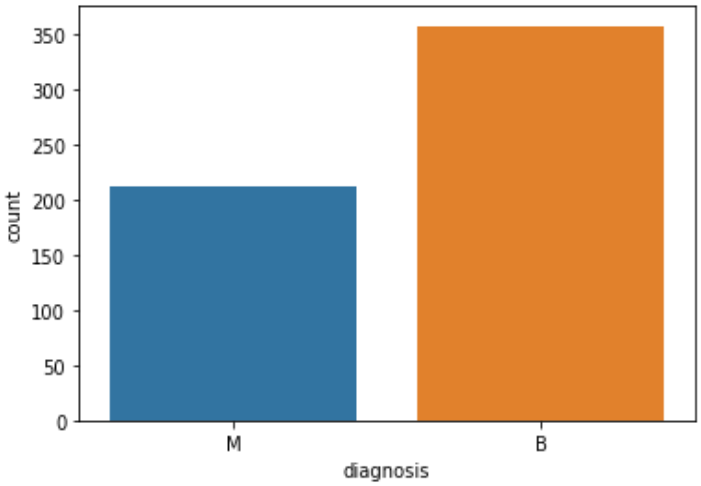
3. **Diagnosis** is our class label.

4. There is an **id** that can not be used for analysis and we drop it.

5. Because differences between values of features are very high to observe on plot (like the **area_mean** feature's max value is 2500 and **smoothness_mean** features' max 0.16340.), so we need standardisation before visualization, feature selection or classification.

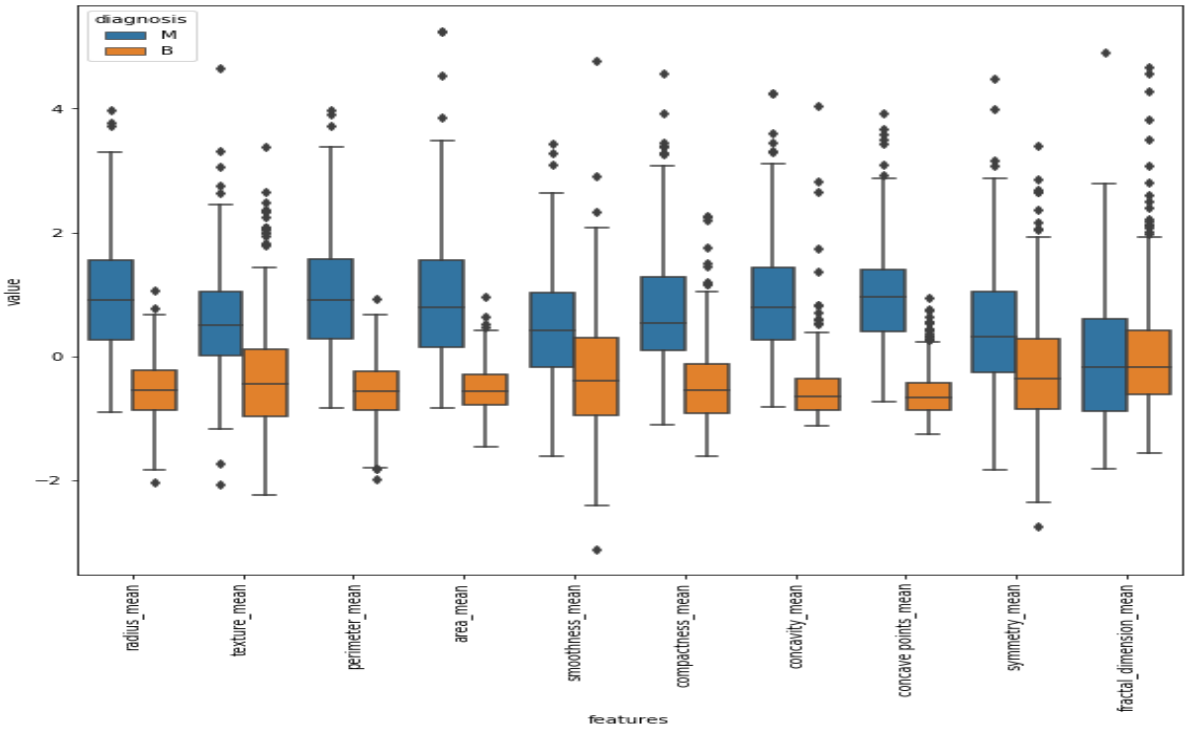
At this point, we do not have any idea about other feature names.

Data Exploration

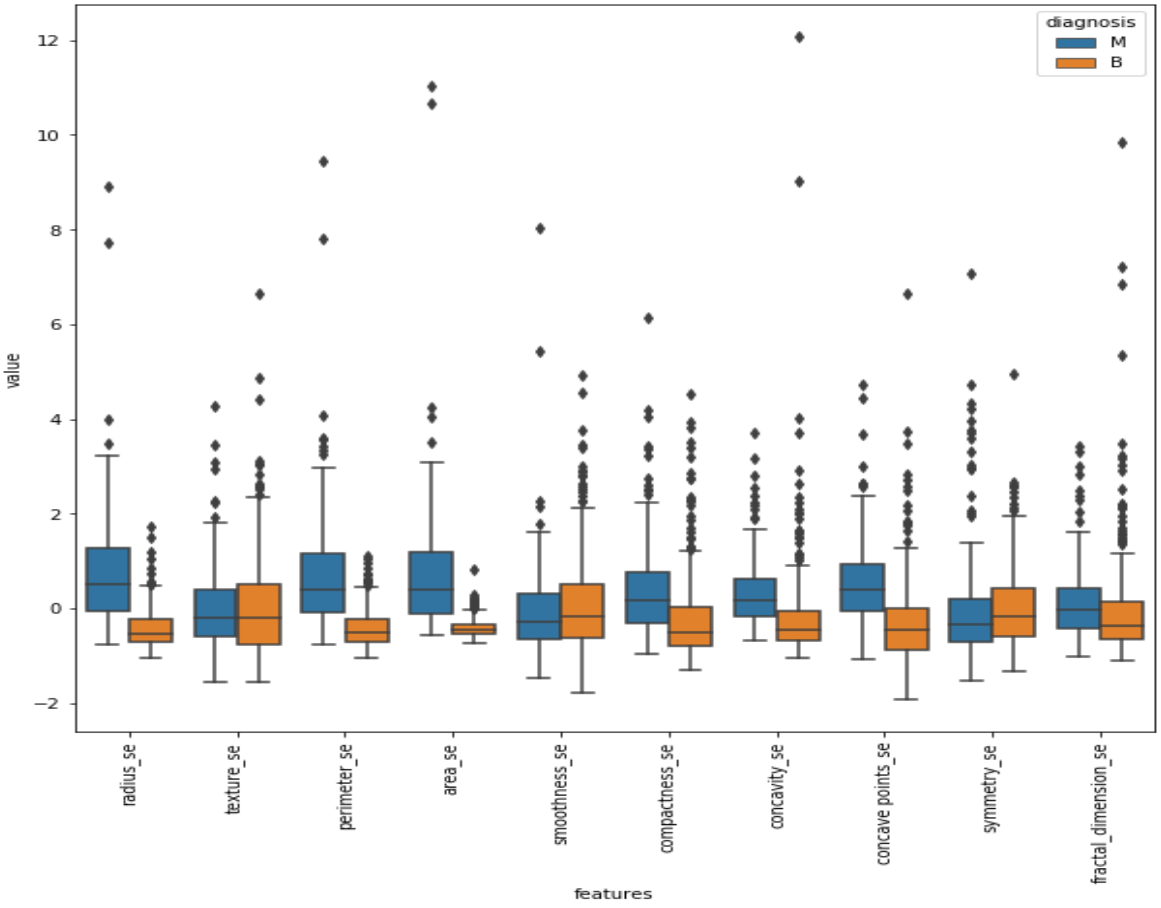



```
Number of Benign:  357
Number of Malignant :  212
```

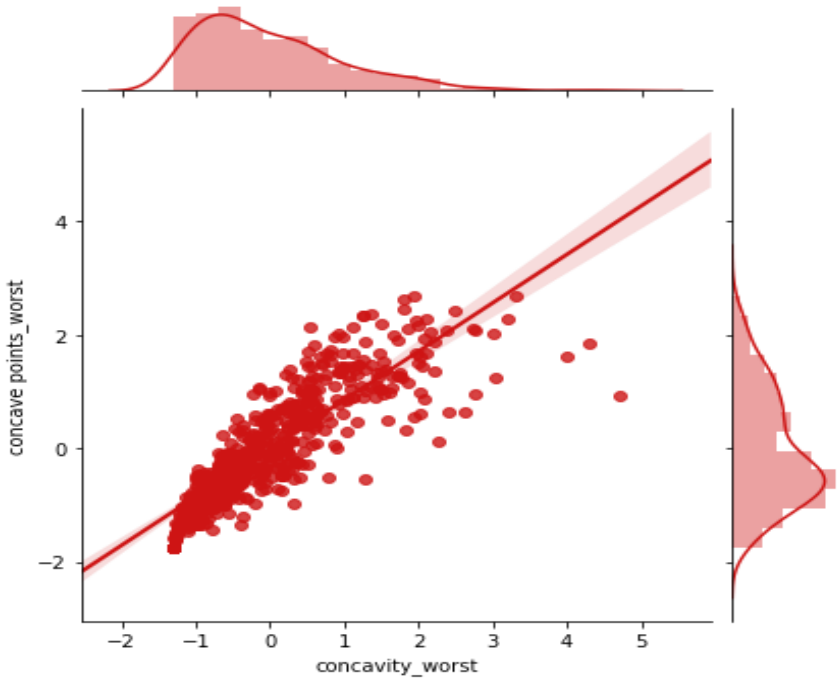
In Box Plot, we plot features in 3 group and each group includes 10 features to observe better.

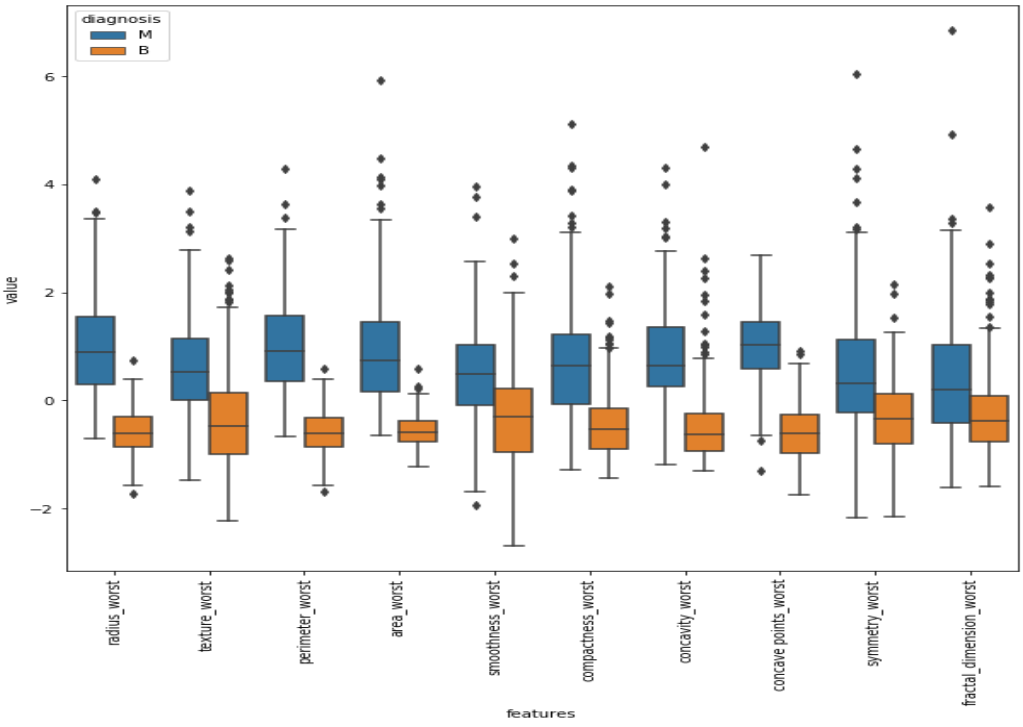


in **texture_mean** feature, median of the *Malignant* and *Benign* looks like separated so it can be good for classification. However, in **fractal_dimension_mean** feature, median of the *Malignant* and *Benign* does not looks like separated so it does not give good information for classification.

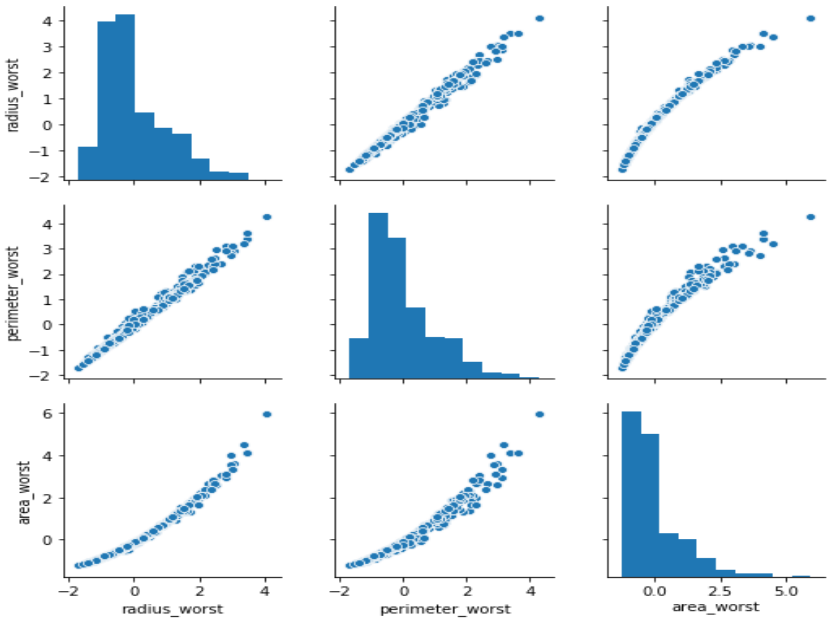


in **concave points_se and concavity_se** features, median of the *Malignant* and *Benign* looks like separated so it can be good for classification. However, in **texture_se** feature, median of the *Malignant* and *Benign* does not looks like separated so it does not give good information for classification. Also looks like there is correlation between **concavity_worst** and concave **point_worst**.

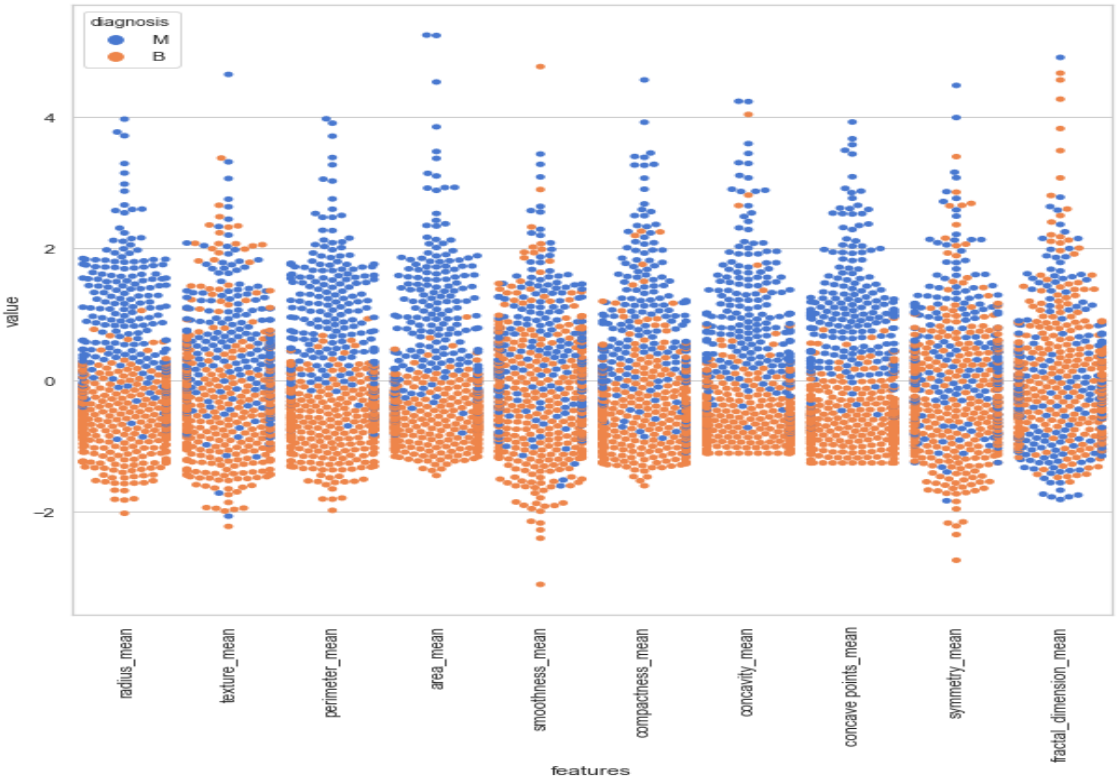




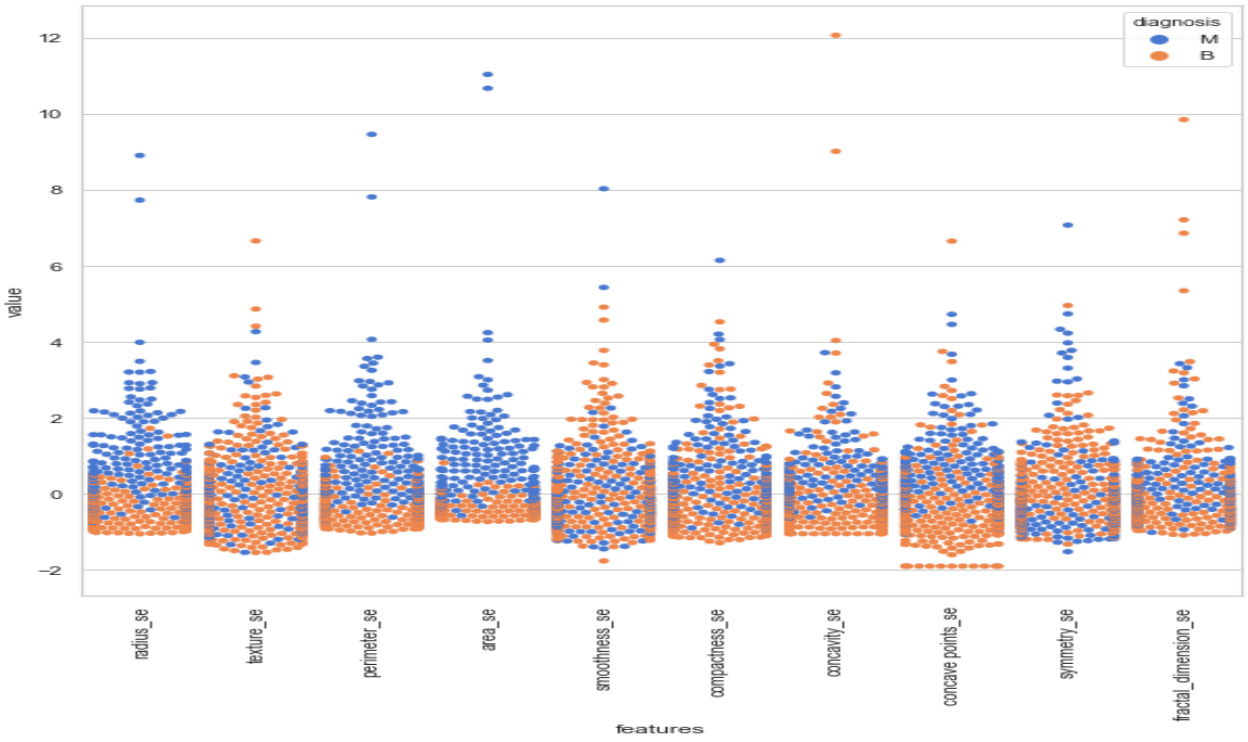
Also it seems **radius_worst**, **perimeter_worst** and **area_worst** are correlated as it can be seen pair plot. We use these for feature selection.



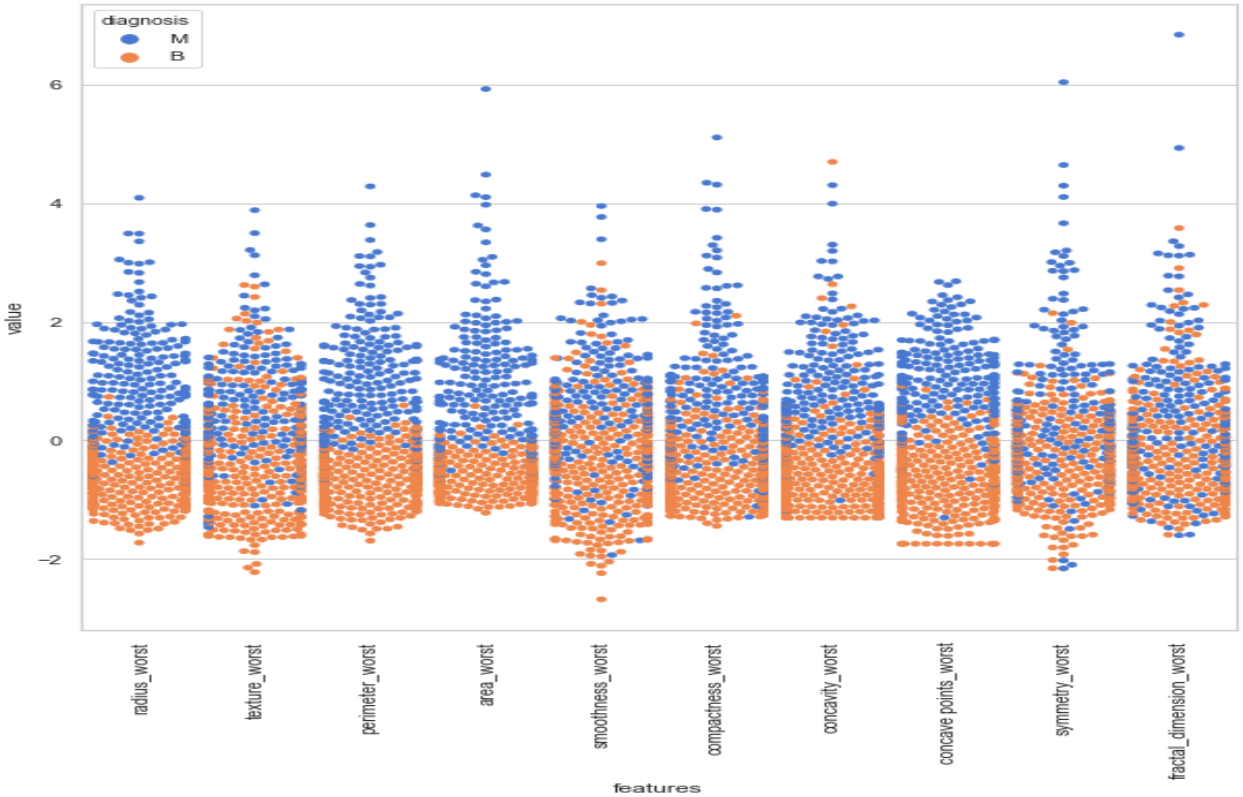
Now in swarm Plot, we plot features in 3 group and each group includes 10 features to observe better.



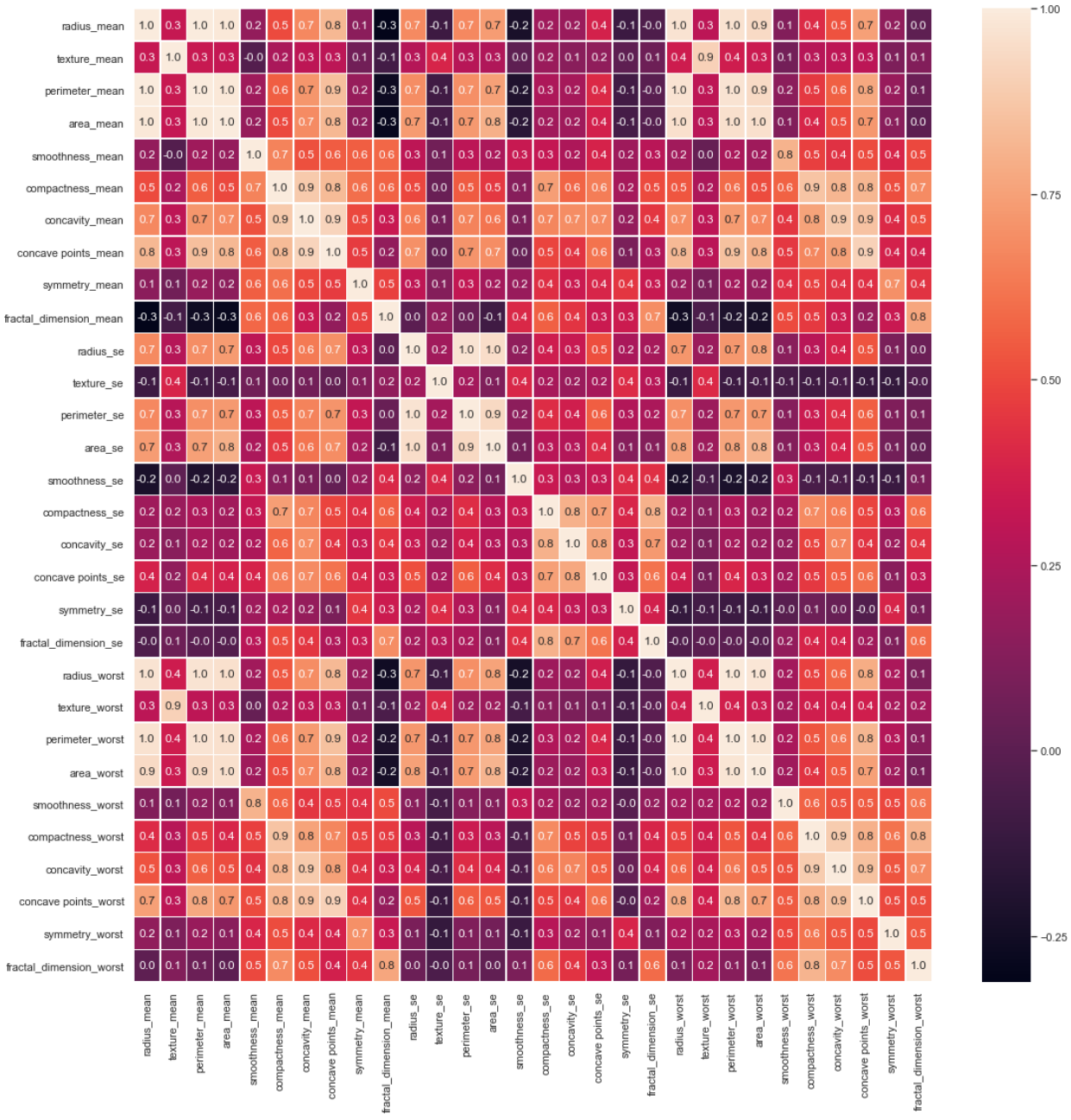
I think **perimeter_mean** and **area_mean** looks like malignant and benign are seprated not totaly but mostly.



I think **area_se** looks like malignant and benign are seprated not totaly but mostly. Hovewer, **smoothness_se** looks like malignant and benign are mixed so it is hard to classfy while using this feature.



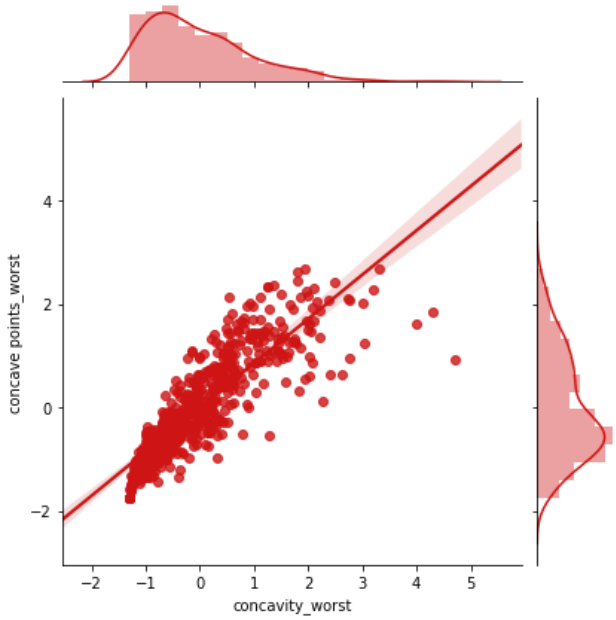
I think area_worst and perimeter_worst looks like malignant and benign are seprated not totaly but mostly. Hovewer, texture_worst looks like malignant and benign are mixed so it is hard to classfy while using this feature.



EDA – Inferential Statistics

In the following EDA, I try to perfom analyze correlation between variable of concavity_worst and concave point_worst.

Initial observations for Correlation between concavity_worst and concave point_worst



Pearson correlation coefficient is 0.86, which means there is strong correlation between these two variables.

Setup hypothesis test

Ho : The variable of concavity_worst and concave point_worst are independent

Ha : The variable of concavity_worst and concave point_worst are correlated

Statistical significance for α = 0.01

Test Statistic: Pearson correlation coefficient

To do so, permute the concavity_worst but leave the concave point_worst values fixed. This simulates the hypothesis that they are totally independent of each other. For each permutation, compute the Pearson correlation coefficient and assess how many of your permutation replicates have a Pearson correlation coefficient greater than the observed one.As p-value less than α, we can reject the null hypothesis and accept alternative hypothesis which means that there is a correlation between variable of concavity_worst and concave point_worst

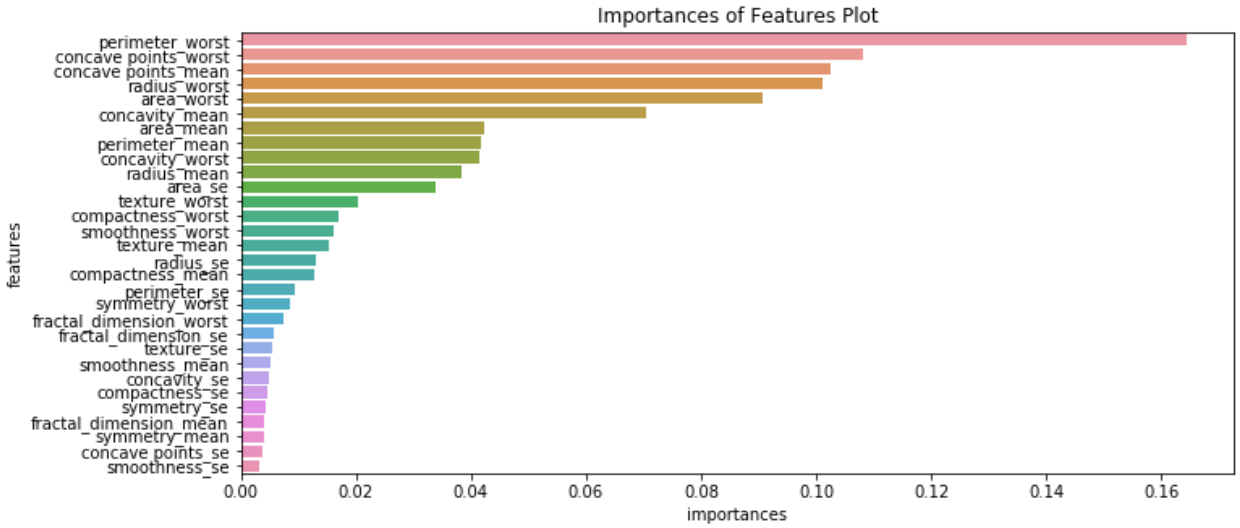
Machine Learning: Overview

Decision Tree is unreliable, because if I change my data a little bit, the decision tree created can be very different. In such a case, I can build a robust model (reduce variance) through bagging like Random Forest. Boosting reduces variance and reduces bias. It reduces variance because I am using multiple models (bagging). It reduces bias by training the subsequent model by telling him what errors the previous models made (the boosting part). So, I took following steps:

- Random Forest (RF) was used to find out importance of features.
- Boruta and Variance Inflation Factor (VIF) used for features selection.
- XGBoost was used for prediction.
- Overall strategy:
    1. Use Random Forest to find out importance of components
    2. Reduce features with VIF compare with Boruta has more accuracy in this case
    3. Use XGBoost on important components of all data.
    4. Split the data into training and test datasets
    5. Train various learning models, with cross-validated hyperparameter tuning
    6. Apply the models to the test dataset to get measures of performance

Random Forest

I visualize the importance of features using random forest model.



Feature Selection with Bruta and VIF

With using Bruta, features has been reduced to 23 features and with using Variance Inflation Factor (VIF) features has been reduced to 13 features.

XGBoost

- The best estimator was selected by Randomized Grid Search with Stratified Shuffle Split Cross Validation.
- The best hyperparameters included a tree count of 250, max depth of 10, and allowing trees to consider all features

Conclusion

The results of the XGBoost algorithm with Boruta feature selection

```
Accuracy is:  0.9298245614035088
```

The results of the XGBoost algorithm with VIF feature selection

```
Accuracy is:  0.935672514619883
```

I achieved almost 94% accuracy in classifying breast cancer samples with combination of VIF and XGBoost method. This model run extremely quick.